Task Allocation for Energy Optimization in Fog Computing Networks With Latency Constraints

Bartosz Kopras, Bartosz Bossy, Filip Idzikowski, *Member, IEEE*, Paweł Kryszkiewicz, *Senior Member, IEEE*, and Hanna Bogucka, *Senior Member, IEEE*

***Abstract*— Fog networks offer computing resources of varying capacities at different distances from end users. A Fog Node (FN) closer to the network edge may have less powerful computing resources compared to the cloud, but the processing of computational tasks in FN limits long-distance transmission. How should the tasks be distributed between fog and cloud nodes? We formulate a universal non-convex Mixed-Integer Nonlinear Programming (MINLP) problem minimizing task transmission- and processing-related energy with delay constraints to answer this question. It is transformed with Successive Convex Approximation (SCA) and decomposed using the primal and dual decomposition techniques. Two practical algorithms called Energy-EFFicient Resource Allocation (EEFFRA) and Low-Complexity (LC)-EEFFRA are proposed and their effectiveness is tested for various network and traffic scenarios. Using EEFFRA/LC-EEFFRA can significantly decrease the number of computational requests with unmet delay requirements when compared with baseline solutions (from 48% to 24% for 10 MB requests). Utilizing Dynamic Voltage and Frequency Scaling (DVFS) minimizes energy consumption (by one-third) while satisfying delay requirements.**

***Index Terms*— Fog network, energy-efficiency, latency, cloud, edge computing.**

I. Introduction

THE number of Internet of Things (IoT) devices exceeds 14 billion worldwide and this number continues to grow [1]. Such devices often have limited memory capacity and computational power. However, processing requests of all IoT devices at a remote cloud would require an unprecedented amount of traffic traversing the Internet [2] and influencing energy consumption as well as congestion of the Internet. Moreover, some applications, such as video surveillance, augmented reality, or vehicle-to-vehicle communication require low delays that cannot be fulfilled by remote cloud Data Centers (DCs). Fog computing [3] addresses these problems by introducing a fog tier (or multiple hierarchical fog tiers) between the *cloud* and the *end devices* tiers (Fig. 1).[1] It is composed of Fog Nodes (FNs) with computational and storage resources located near end users. Data transmission between *end devices* and FNs is thus faster and potentially less energy-consuming than alternative thing-to-cloud communication.

Manuscript received 17 January 2022; revised 14 June 2022 and 6 October 2022; accepted 11 October 2022. Date of publication 25 October 2022; date of current version 19 December 2022. The presented work has been funded by the Polish Ministry of Education and Science within the research bailout for task "Transmission and reception of radio signals, design and testing of radio communication circuits and networks" in 2022 and by the National Science Centre in Poland within the FitNets project no. 2021/41/N/ST7/03941 on "Fresh and Green Cellular IoT Edge Computing Networks — FitNets." The associate editor coordinating the review of this article and approving it for publication was N. Pappas. *(Corresponding author: Filip Idzikowski.)*

The authors are with the Faculty of Computing and Telecommunications, Poznan University of Technology, 60-965 Poznan, Poland (e-mail: filip.idzikowski@put.poznan.pl).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2022.3216645.

Digital Object Identifier 10.1109/TCOMM.2022.3216645

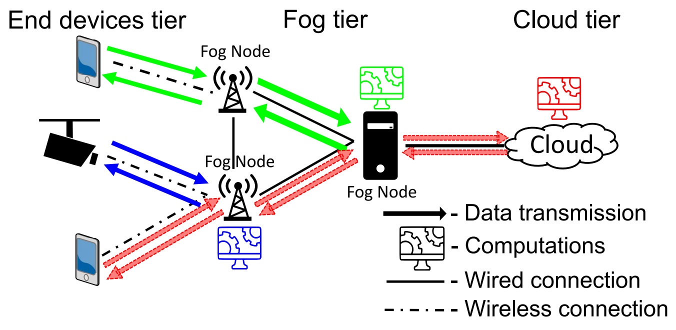


Fig. 1. Example of fog computing network with task offloading scenario (colors differentiate offloaded tasks).

This work considers task distribution between many FNs and Cloud Nodes (CNs). We minimize network energy consumption while meeting delay constraints specific for each offloaded task. As shown in Fig. 1, an offloaded task can be processed in the node to which it is originally sent (solid blue arrows), in another FN (solid green arrows), or in the cloud (hollow red arrows). A realistic network model is proposed below encompassing the energy consumption as well as the delay, related to both the necessary computations and communication. Our model includes realistic network parameters reflecting the characteristics of real-world equipment. Based on this model, we formulate an optimization problem to minimize the total (task transmission- and computation-related) energy consumption while fulfilling total delay constraints. The optimization considers not only the assignment of tasks to the nodes but also the Central Processing Unit (CPU) frequency at each utilized node. The problem is a non-convex Mixed-Integer Nonlinear Programming (MINLP) problem, so we apply the Successive Convex Approximation (SCA) method which transforms it into a series of convex

[1] *Fog computing* is closely related to *edge computing*, but we distinguish *the fog* from *the edge* by the mentioned multiple hierarchical layers of the fog and flexibility of directing the computational tasks to suitable FNs introduced by such architecture.

MINLP problems, and provides the optimal solution by using the primal and dual decomposition techniques as well as the Hungarian algorithm. A sub-optimal, lower complexity solution is also proposed. The main contributions of our work include: 1. modeling energy consumption and delays related to transmission, queuing, and computing of tasks in the fog and cloud tiers, 2. proposing and solving a complex optimization problem, 3. examining the efficiency of proposed solutions for a vast scope of network and traffic parameters. Our solutions are original in handling more complex network models, more realistic parameters, and considering both task processing- (computing-) and transmission-related energy and delay, as opposed to the background works.

The paper is organized as follows. An overview of related work is provided in Section II. A mathematical model of the fog network is presented in Section III. Section IV defines the optimization problem, and Section V presents its solution. Simulation results for various scenarios are shown and discussed in Section VI. Section VII concludes the work.

II. Related Work

In this Section, we review related work while focusing on the following aspects: (i) energy consumption of fog networks related to task implementation (computations) and transmission of computational tasks, (ii) consideration of latency caused by both computations and transmission, (iii) fog scenarios, network and traffic models, (iv) optimization of energy consumption. The aspects (i)–(iv) stress the novelty of our work, where we jointly study communication and computing in the fog in the context of offloading computational tasks. Works which focus on other potential applications of fog computing (e.g., content distribution/caching [4], [5], and [6]) are therefore not included in this overview as their network and traffic scenarios are significantly different from ours. Papers not taking energy consumption into account, e.g., [7], [8], and [9], and those that only look at power consumption as a constraint, e.g., [10] are also intentionally omitted.

The authors of [11], [12], and [13] look at the energy consumption of individual end devices. They aim to answer the following question: is it more efficient for an end device to process a task locally, or to offload it to the fog/cloud? These works disregard the energy costs related to performing computations and transmission in the fog or cloud tiers of the network (our aspect (i)) and are therefore substantially different from our work. Our work and those we survey further below examine total energy consumption from the network operator's point of view, tackling the following question: given a task has already been offloaded, where is it advantageous to process it?

Sarkar et al. examine the energy consumption [14], power consumption [15], and task delay [14], [15] in the fog-computing network, depending on how much data is processed in the fog tier and in the cloud tier. This data is transmitted from terminal nodes. Their models include costs related to transmission, processing, and storage of data. These works show that both power/energy consumption and delay decrease with a higher percentage of tasks being processed in FNs rather than the cloud. Power/energy consumption and delay also increase with the number of terminal nodes. However, this increase is not monotonous. Moreover, a discussion on application-specific fog computing utilization is included in [14], while costs related to the carbon footprint of the network are examined in [15].

There are multiple major differences between our work and [14], [15]. First, there is no optimization (aspect (iv)) in these works. While costs are modeled depending on where offloaded requests are sent, no solution for their optimization is proposed. Second, FNs in [14] and [15] work at a fixed clock frequency, while in our work, DVFS is considered (aspect (iii)). Furthermore, costs related to transmission within the fog tier (aspects (i) and (ii)) are also not considered in [14] and [15]. Finally, peculiar assumptions (indefinite processing of data in cloud DCs [15], energy dissipation rate defined as a sum of energy spent on computations over time plus an average of energy spent on transmission over time [14]) and mistakes (e.g., triangle inequality used incorrectly in Sec. 5.2.1 of [15]) make results of [14], [15] biased towards showing that fog computing is significantly faster and more energy-efficient than cloud computing regardless of the offloading scenario.

In [16], offloaded requests can be served by one of the FNs or the cloud servers. Similar to our work, the objective of [16] is to minimize the power consumption of the network while maintaining delay constraints. Delay related to queuing and computing is calculated as an average response time of queuing models (M/M/1 in FNs and M/M/n in DCs). The cloud servers can adjust their clock frequency using DVFS. The results show a clear trade-off between power consumption and delay. The aspects (i), (iii), and (iv) distinguish [16] from our work. The energy/power costs are not related to data transmission in [16] (i). Also, rather than jointly optimizing energy costs related to task transmission and processing (computations) in the fog and the cloud, Deng et al. [16] heuristically split this optimization problem into three subproblems (iv). Offloaded traffic is not divided into a number of requests, packets, or instances in [16] (iii) and is only parameterized with a single number. As a consequence, only the average delay can be calculated (and constraint satisfied), rather than the individual delay of each offloaded request. In this work, traffic offloaded from end devices consists of multiple requests, where each request is defined by its size, arithmetic intensity, and delay requirement.

Vakilian et al. [17], [18] jointly optimize delay and energy consumption related to offloading in fog networks with multiple FNs. FNs can cooperate by sending workload to each other and the cloud. Similarly to [16], delay related to queuing and computing is calculated as an average response time of the queuing model (M/M/1 for workload processed in FNs) with a constant value added for transmission delay between FNs. While both works use similar objective functions (weighted sum of energy consumption and delay), [18] includes *fairness coefficients* that depend on resources available to FNs while [17] does not. The authors conclude that the problem in [17] is convex. For [18], they propose a population-based algorithm, i.e., the cuckoo evolution algorithm. The results of [17] show a clear trade-off between energy consumption and delay in the network. The method proposed in [18] decreases both delay and energy consumption compared with a competing algorithm (proposed in [19]), while both works highlight that cooperation between nodes leads to lowered costs.

Vakilian et al. [17], [18] do not consider adjusting CPU frequencies of nodes. Furthermore, offloaded traffic is not divided into a number of requests, packets, or instances in [17] and [18]. It is only parameterized with a single number (like in [16]). Hence, [17], [18] differ from our work in aspect (iii). Finally, joint energy and latency minimization is performed in [17] and [18], while energy minimization under latency constraints is performed in our work – aspect (iv).

Cai et al. [20] examine a network with a single task node and multiple helper nodes which lack an external power supply. The task node can process computational tasks itself or offload these tasks to the helper nodes. To do so, it needs to transfer both the task and energy required for computations to the helper node. The authors of [20] optimize cost defined as a weighted sum of delay and energy consumed by the task node for computations, energy transmission, and task transmission. The constraints include a maximum task execution delay that cannot be exceeded and transferring enough energy for computations in helper nodes. Optimization algorithms are proposed for scenarios with and without the possibility of queuing tasks.

There are multiple differences between [20] and our work. In terms of energy consumption (i), our work minimizes the energy consumed by all computing nodes, while [20] minimizes the energy spent only by the task node, energy consumption of helper nodes is examined but only as a constraint. There are significant differences in network and traffic models (iii). The authors of [20] examine an energy harvesting scheme where helping nodes are gated by the amount of energy they receive. In [20] tasks originate from one task node while in our work they arrive from end devices at any of our FNs. In contrast to our work, all nodes in [20] work at fixed clock frequencies without any connection to the cloud.

Power consumption and delay in fog and cloud tiers of fog computing networks are studied in [21]. The results are shown for various traffic and network parameters. A trade-off between power consumption and delay is shown both in the number of FNs and their clock frequency (more FNs and higher frequencies mean higher power consumption and lower delay). However, in [21], similarly to [14] and [15], power consumption and delay in the fog computing network are only examined – there is no energy-cost optimization (aspect (iv)). Also, while delay and energy costs related to transmission between fog and cloud tiers are considered in [21], the cost related to transmission within the fog tier is not considered (aspects (i) and (ii)).

To summarize, we minimize the energy consumed by both networking and computing equipment in the fog computing network. To the best of our knowledge, no prior work tackles the problem of jointly minimizing energy spent on both transmission and computation by distributing tasks between the fog and the cloud, while satisfying latency constraints and dynamically choosing optimal CPU clock frequencies. Finally, we take the energy- and latency-costs of inter-FN communication into account in our optimization algorithms, while they are ignored or assumed to be negligible in the surveyed papers.

III. Network Model

We introduce the network model in this section. Notation is presented in Table I. Letters in superscript are used throughout this work as upper indices, not exponents, e.g., L^r does **not** denote L to the power of r.

In the bottom tier of the network, there are end devices (e.g., smartphones, sensors) with specific computational tasks. We assume that serving these tasks requires offloading them, i.e., they either cannot be processed in the end device or the end device chooses to offload them rather than to process them locally. Then, they can be processed either in the fog tier, consisting of set $\mathcal{F}$ of FNs, or in the cloud tier (set $\mathcal{C}$ of DCs). A set of all computing nodes in the network is denoted as $\mathcal{N} = \mathcal{F} \cup \mathcal{C}$.

Unlike works that focus on end devices [11], [12], and [13], we examine energy consumption from the point of view of the fog network. Modeling and optimizing wireless transmission is a key part of these works, e.g., allocating sub-channels to mobile devices in [12] or interference affecting transmission rates in [13]. Our system model is agnostic about the model of the end device-FN link. However, depending on the chosen technology, the number of devices accessing the medium, and data volume, the communication delay can vary. This is considered in the presented model, as it can influence the feasibility of a given task allocation strategy. Meanwhile, the energy spent on wireless transmission between an end device and FN is independent from the task allocation strategy used by the fog network. Therefore, we do not include it in the optimization problem. We focus on the efficient distribution of offloaded tasks between fog and cloud nodes.

A. Computational Requests

Let $\mathcal{T}$ be a numbered set $\{T_1, T_2, \ldots, T_{|\mathcal{T}|}\}$ of all time instances at which computational requests arrive at FNs, and have to be allocated computing resources. Let R_k be a set of all requests arriving in the network at time T_k. Each computational request $r \in R_k$ is described by the following parameters: (i) size L^r in bits, (ii) arithmetic intensity θ^r in FLOP/bit (used in FLOP/byte in [22] and [23]), (iii) ratio o^r of the size of the result to the size of the offloaded task (most related works do not consider the transmission of the results [14], [15] or assume that its contribution is negligible [16], [20]; o^r equal to 1 implies the output has the same size as the input; in case of electrocardiography signals $o^r \simeq 0.07$ [24]), (iv) FN $g^r \in \mathcal{F}$ to which the request is originally sent (before allocation), (v) maximum tolerated delay D^r_{max}, (vi) time of request creation in the end device t^{orgin}_r, (vii) delay introduced by task offloading from the end device to an edge node τ^r_{req} where $t^{\text{orgin}}_r + \tau^r_{\text{req}} = T_k$, and (viii) delay introduced by transmitting response (results) from

TABLE I
NOTATION FOR MODELING FOG COMPUTING NETWORK AND DEFINING OPTIMIZATION PROBLEM

	Symbol	Description
Parameters	b_{back}	link bitrate in the backhaul and backbone network
	b_r^n	link bitrate between FNs $n \in \mathcal{F}$ and $g^r \in \mathcal{F}$
	$\mathcal{C}$	set of all cloud DCs
	χ	parameter characterizing delay depending on distance
	d^n	fiberline distance from the fog to cloud DC $n \in \mathcal{C}$
	D^r_{max}	maximum tolerated delay requirement for request $r \in R_k$
	$\mathcal{F}$	set of all Fog Nodes
	$f_{\text{max},n}$	maximum clock frequency of node $n \in \mathcal{N}$
	$f_{\text{min},n}$	minimum clock frequency of node $n \in \mathcal{N}$
	g^r	FN to which the request $r \in R_k$ is originally sent
	γ_n^r	energy-per-bit cost of transmitting data of request $r \in R_k$ between nodes $g^r \in \mathcal{F}$ and $n \in \mathcal{N}$
	L^r	size of request $r \in R_k$
	$\mathcal{N}$	set of all nodes
	o^r	output-to-input data size ratio of request $r \in R_k$
	$p_{n,q}$	q-th coefficient of polynomial modeling power consumption of CPU installed in node $n \in \mathcal{N}$
	Q	degree of polynomial P_n
	R_k	set of all computational requests offloaded at T_k
	R_k'	set of rejected computational requests offloaded at T_k
	s_n	number of FLOPs performed per single clock cycle at node $n \in \mathcal{N}$
	$\mathcal{T}$	set of all considered time instances, when one or more computational requests arrive
	θ^r	arithmetic intensity of request $r \in R_k$
	T_k	time at which request $r \in R_k$ arrives in the network, $k \in \{1, ..., \lvert\mathcal{T}\rvert\}$
	$t_{n,k}$	time at which node $n \in \mathcal{F}$ is scheduled to finish computing its last task when requests arrive at T_k
	t_r^{orgin}	time of request $r \in R_k$ creation in the end device
	τ^r_{req}	delay introduced by offloading task $r \in R_k$ from the end device to FN $g^r \in \mathcal{F}$
	τ^r_{res}	delay introduced by transmitting results of request $r \in R_k$ from FN $g^r \in \mathcal{F}$ to the end device
Variables and metrics	a_n^r	variable showing whether request $r \in R_k$ is computed at node $n \in \mathcal{N}$, $a_n^r \in \{0, 1\}$
	β_n	energy efficiency (FLOPS per Watt) characterizing node $n \in \mathcal{N}$
	$D^r_{\text{wd},n}$	delay caused by transmitting request $r \in R_k$ between nodes $g^r \in \mathcal{F}$ and $n \in \mathcal{N}$
	$D^r_{\text{wddl},n}$	delay caused by transmitting request $r \in R_k$ between nodes $g^r \in \mathcal{F}$ and $n \in \mathcal{N}$ – downlink
	$D^r_{\text{wdul},n}$	delay caused by transmitting request $r \in R_k$ between nodes $g^r \in \mathcal{F}$ and $n \in \mathcal{N}$ – uplink
	D^r_{cp}	computational delay caused by processing request $r \in R_k$ in the network
	$D^r_{\text{cp},n}$	computational delay caused by processing request $r \in R_k$ at node $n \in \mathcal{N}$
	D^r_{max}	maximum tolerated delay requirement for request $r \in R_k$
	$D^r_{\text{queue},n}$	queuing delay of request $r \in R_k$ at node $n \in \mathcal{N}$
	D^r_{tot}	total delay of request $r \in R_k$
	$D^r_{\text{tot},n}$	total delay of processing request $r \in R_k$ at node $n \in \mathcal{N}$
	E^r_{comm}	energy spent on transmission of request $r \in R_k$
	$E^r_{\text{comm},n}$	energy cost for transmission of request $r \in R_k$ between nodes $g^r \in \mathcal{F}$ and $n \in \mathcal{N}$
	E^r_{cp}	energy spent in the network on processing request $r \in R_k$
	$E^r_{\text{cp},n}$	energy cost of processing request $r \in R_k$ at node $n \in \mathcal{N}$
	E^r_{tot}	energy spent on transmission and processing of request $r \in R_k$
	$E^r_{\text{tot},n}$	energy cost of offloading request $r \in R_k$ when computing it at node $n \in \mathcal{N}$
	f_n	clock frequency of node $n \in \mathcal{N}$
	P_n	power consumption related to computations at node $n \in \mathcal{N}$

FN g^r to the end device τ^r_{res}. Let us define a binary variable a_n^r that shows where the request is computed, i.e., a_n^r equals 1 if $r \in R_k$ is computed at node $n \in \mathcal{N}$, and 0 otherwise.

B. Energy Consumption

The energy consumption model consists of two parts: communication (transmission of data) and computation (processing of data). Energy E^r_{cp} spent on processing request $r \in R_k$ equals:

$$E^r_{\text{cp}} = \sum_{n \in \mathcal{N}} a_n^r E^r_{\text{cp},n} = \sum_{n \in \mathcal{N}} a_n^r \frac{L^r \theta^r}{\beta_n}, \tag{1}$$

where $E^r_{\text{cp},n}$ is the energy spent on processing request $r \in R_k$ at node $n \in \mathcal{N}$. β_n characterizes the computational efficiency of node $n \in \mathcal{N}$ given in Floating Point Operations (FLOPs) per second per watt [25]. For cloud DCs, we assume constant CPU clock frequency f_n and efficiency β_n. For FNs, β_n depends on CPU clock frequency f_n of node $n \in \mathcal{F}$, its power consumption P_n, and number s_n of FLOPs performed within a single clock cycle of this node [26]:

$$\beta_n = \frac{f_n s_n}{P_n} = \frac{f_n s_n}{\sum_{q=0}^{Q} p_{n,q} f_n^q}. \tag{2}$$

We model P_n as a Q-th degree polynomial of f_n using parameters $p_{n,q}$ based on [27]. This allows the model to cover various CPUs. Moreover, clock frequency f_n must be within the range of minimum and maximum frequencies of the CPU installed in node $n \in \mathcal{F}$, i.e., $f_{\text{min},n} \leq f_n \leq f_{\text{max},n}$.

The energy spent on the transmission of request $r \in R_k$ equals:

$$E^r_{\text{comm}} = \sum_{n \in \mathcal{N}} a_n^r E^r_{\text{comm},n} = \sum_{n \in \mathcal{N}} a_n^r L^r (1 + o^r) \gamma_n^r, \tag{3}$$

where $E^r_{\text{comm},n}$ is the energy required to transmit (communicate) request $r \in R_k$ between FN g^r and node $n \in \mathcal{N}$ while γ_n^r is the energy-per-bit cost of transmitting data request $r \in R_k$ between node n and node g^r. The energy required for communication between the network edge and the end device does not influence the task allocation result and is therefore omitted.

$L^r o^r$ is the size (in bits) of results transmitted back to FN g^r. Thus, the total energy spent on offloading request $r \in R_k$ is given by

$$E^r_{\text{tot}} = \sum_{n \in \mathcal{N}} a_n^r E^r_{\text{tot},n} = \sum_{n \in \mathcal{N}} a_n^r \left(E^r_{\text{cp},n} + E^r_{\text{comm},n}\right), \tag{4}$$

where $E^r_{\text{tot},n}$ is the energy cost of offloading request $r \in R_k$ when it is computed at node $n \in \mathcal{N}$.

C. Delay

The delay model is divided into four parts which are discussed in the following subsections.

1) Device-to-Fog Communication: The delay on the link between the end device and FN can use various technologies and be more or less prone to delay variations based on, e.g., interference. However, as the delay τ^r_{req} is observed before the task allocation decision happens at T_k, its value is perfectly known. Moreover, it has the same influence on any possible allocation option. The response time τ^r_{res} is more difficult to model. While the response time can be a random variable, it is reasonable to use, e.g., the 98th percentile of possible delay for τ^r_{res}. Such a solution (assuming a near worst-case scenario) assures that the results of computations arrive at the end node within the tolerated delay.

2) Fog-to-Fog and Fog-to-Cloud Communication: There are significant differences between models of delay for requests processed in the fog tier and the cloud tier of the network. It stems from the fact that clouds are assumed to have huge (practically infinite) computational resources with parallel-computing capabilities, and there is no need for queuing multiple requests served by the cloud DC $n \in \mathcal{C}$. They can be processed simultaneously. Meanwhile, if multiple requests are sent to the same FN $n \in \mathcal{F}$ for processing in a short time span, additional delays may occur due to the congestion of computational requests (an arriving request cannot be processed until processing of all previous requests has been completed). On the other hand, it is assumed that cloud DCs are located far away from the rest of the network (hundreds or even thousands of kilometers away) which introduces additional, transmission-related delays. Delay caused by transmitting request $r \in R_k$ between (to and from) FN $g^r \in \mathcal{F}$ and cloud node $n \in \mathcal{C}$ is equal:

$$D^r_{\text{comm},n} = \frac{L^r(1+o^r)}{b_{\text{back}}} + 2d^n \cdot \chi, \tag{5}$$

where b_{back} is the link bitrate in the backhaul and backbone network, while d^n is the fiberline distance to cloud DC $n \in \mathcal{C}$. Parameter χ indicates the rate at which the delay increases with distance d^n [28].

For describing delays related to transmission between FNs let us split it into the uplink (sending a request to be processed) and downlink (sending calculated results back to the origin of said request) parts denoted $D^r_{\text{commUL},n}$ and $D^r_{\text{commDL},n}$ respectively. For transmission between FNs, we assume the delay caused by the distance between them ($2d^n \cdot \chi$ in Eq. (5)) to be negligible – well below 1 ms as we use the value of 7.5μs/km for parameter χ [28] – and therefore we ignore it. The total delay caused by communication between FN $g^r \in \mathcal{F}$ and $n \in \mathcal{F}$ for request $r \in R_k$ equals:

$$D^r_{\text{comm},n} = D^r_{\text{commUL},n} + D^r_{\text{commDL},n} = \frac{L^r}{b^n_r} + \frac{L^r o^r}{b^n_r}, \tag{6}$$

where b^n_r is the link bitrate between FN g^r and n.

3) Queuing: As discussed earlier, when a request is sent to FN $n \in \mathcal{F}$ and there is another request being processed at this node, the request is put in a queue and waits to be processed. Let us define a scheduling variable $t_{n,k} \in \mathbb{R}^+$ which indicates when the processing of the last request scheduled at FN $n \in \mathcal{F}$ is completed. The queuing delay of request $r \in R_k$ at node $n \in \mathcal{F}$ is calculated as follows:

$$D^r_{\text{queue},n} = \max(0, t_{n,k} - T_k - D^r_{\text{commUL},n}). \tag{7}$$

$D^r_{\text{queue},n}$ has positive values when $t_{n,k} > T_k + D^r_{\text{commUL},n}$, i.e., when request r arrives at node n at time $T_k + D^r_{\text{commUL},n}$ and it is queued until the processing of other request(s) is completed at time $t_{n,k}$. For each node $n \in \mathcal{C}$ (cloud DCs), $D^r_{\text{queue},n}$ is always equal to zero, i.e., each request arriving at the cloud can immediately be processed regardless of the number of requests already being processed due to parallel processing.

4) Computations: Delay D^r_{cp} caused by computing request $r \in R_k$ equals:

$$D^r_{\text{cp}} = \sum_{n \in \mathcal{N}} a^r_n D^r_{\text{cp},n} = \sum_{n \in \mathcal{N}} a^r_n \frac{L^r \theta^r}{f_n s_n}, \tag{8}$$

where $D^r_{\text{cp},n}$ is the time required to compute request $r \in R_k$ at node $n \in \mathcal{N}$.

Thus, total delay D^r_{tot} of processing request $r \in R_k$ is the sum of delays related to transmission, queuing, and computation:

$$\begin{aligned} D^r_{\text{tot}} &= \sum_{n \in \mathcal{N}} a^r_n D^r_{\text{tot},n} \\ &= \sum_{n \in \mathcal{N}} a^r_n \left(\tau^r_{\text{req}} + D^r_{\text{comm},n} + D^r_{\text{queue},n} + D^r_{\text{cp},n} + \tau^r_{\text{res}}\right), \end{aligned} \tag{9}$$

where $D^r_{\text{tot},n}$ is the total delay of processing request $r \in R_k$ when it is computed at node $n \in \mathcal{N}$.

D. Updating Scheduling Variables in the Fog

Let us now explain how the values of scheduling variables $t_{n,k}$ are assigned to become parameters of an optimization instance. As no requests are processed at the beginning of the simulation, we set $t_{n,k} = 0, \forall n \in \mathcal{F}$. For each $T_k \in \mathcal{T}$, after allocations a^r_n are determined, times $t_{n,k+1}$ are calculated according to when computations of requests offloaded to FNs are scheduled to finish:

$$\begin{aligned} t_{n,k+1} := \max(&t_{n,k}, T_k \\ &+ \sum_{r \in R_k} a^r_n (D^r_{\text{commUL},n} + D^r_{\text{queue},n} + D^r_{\text{cp},n})), \ \forall n \in \mathcal{F}. \end{aligned} \tag{10}$$

By using (10), each new instance of the optimization problem (when new requests arrive at T_{k+1}) depends on results (allocations) of previous instances.

IV. Optimization Problem

The objective of our formulated problem is to minimize the total energy spent on offloading all requests arriving at the network edge at time T_k, that is to find:

$$(\mathbf{a}^\star, \mathbf{f}^\star) = \arg\min_{\mathbf{a},\mathbf{f}} \sum_{r \in R_k} E^r_{\text{tot}}, \tag{11}$$

subject to:

$$\sum_{n \in \mathcal{N}} a^r_n = 1 \quad \forall r \in R_k, \tag{12}$$

$$\sum_{r \in R_k} a^r_n \le 1, \quad \forall n \in \mathcal{F}, \tag{13}$$

$$D^r_{\text{tot}} \le D^r_{\text{max}}, \quad \forall r \in R_k, \tag{14}$$

$$f_{\text{min},n} \le f_n \le f_{\text{max},n} \quad \forall n \in \mathcal{F}, \tag{15}$$

$$a^r_n \in \{0,1\}, \quad \forall r \in R_k, \forall n \in \mathcal{N}, \tag{16}$$

where $\mathbf{a}^\star = \{a^{r\star}_n\}$ and $\mathbf{f}^\star = \{f^\star_n\}$ are optimal values of the optimization variables: allocation variables a^r_n and CPU clock frequencies f_n, respectively. Sets of constraints (12) and (13) restrict that each request must be processed at one and only one FN or cloud DC and that each FN can process at most a single request at a time, respectively. The set of constraints (14) guarantee that the total delay D^r_{tot} must not be greater than the maximum acceptable one $D^r_{\max}$. Moreover, according to the set of constraints (15), the CPU frequency is limited by lower and upper bounds, while decision variables a^r_n take only binary values, according to the set of constraints (16).

The optimization problem cannot be solved for some sets of requests R_k, where it is impossible to satisfy all the constraints (e.g., there is no feasible allocation of requests, so that each request is processed (12) while fulfilling its delay requirement (14)). In this case, rather than terminating the optimization without any solution (which would translate to rejecting all requests R_k), we choose to reject requests for which (14) cannot be fulfilled. The optimization is then performed over the set of remaining requests $R_k \setminus R'_k$, where R'_k denotes the set of rejected requests.

V. Proposed Solution

The optimization problem defined in Section IV is a MINLP problem due to binary values of allocation variables and continuous values of CPU clock frequencies. Nonlinearity results from the power consumption model of the CPU and the set of constraints (14). Our problem as a MINLP problem is NP-hard [29]. However, as all its variables are bounded, it is computable, i.e., there exists a Turing machine that can compute it [29]. Note that after substituting (2) into (1), the energy spent on processing of request $r \in R_k$ at node $n \in \mathcal{F}$ is the sum of polynomial and rational functions:

$$E^r_{\text{cp},n} = \frac{L^r\theta^r}{s_n}\left[\frac{p_{n,0}}{f_n} + \sum_{q=1}^{Q} p_{n,q} f^{q-1}\right]. \quad (17)$$

As such, for $f_n \in \mathbb{R}^+$, the convexity of (17) in f_n depends on parameters $p_{n,q}$ (except $p_{n,1}$ and $p_{n,2}$ which have no influence on convexity, since their second derivatives are zero). If $\{p_{n,0}, p_{n,3}, \ldots, p_{n,Q}\}$ are positive, the objective function is convex. If all these parameters are negative the function is concave. In these cases, standard optimization methods can be used to solve it [30]. However, if some of these parameters are negative, others being positive, we deal with a difference of convex functions which is non-convex, requiring special optimization techniques. Therefore, in this section, the solution to the optimization problem, in the case of any possible values of CPU power consumption parameters is presented as follows.

Let us rewrite the objective function (11) with $E^r_{\text{cp},n}$ being a difference of convex functions:

$$(\mathbf{a}^\star, \mathbf{f}^\star) = \arg\min_{\mathbf{a},\mathbf{f}} \sum_{r\in R_k}\sum_{n\in\mathcal{F}} a^r_n \left(\underbrace{E^{r+}_{\text{cp},n} - E^{r-}_{\text{cp},n}}_{E^r_{\text{cp},n}} + E^r_{\text{comm},n}\right), \quad (18)$$

where $E^{r+}_{\text{cp},n}$ denotes the components of sum $E^r_{\text{cp},n}$ with positive parameters $p_{n,q}$ and $E^{r-}_{\text{cp},n}$ is the negative of the components of sum $E^r_{\text{cp},n}$ with negative parameters $p_{n,q}$. We apply the SCA method [31], [32], and [33] to approximate the possibly non-convex function by a series of convex ones. Since the objective function (18) is composed of differences of convex functions, the subtrahend $E^{r-}_{\text{cp},n}$ can be approximated with a linear function using the first-order Taylor series expansion at $\bar{\mathbf{f}} = \{\bar{f}_n\}$:

$$E^{r-}_{\text{cp},n}(f_n) \le E^{r-}_{\text{cp},n}(\bar{f}_n) + \left.\frac{\partial E^{r-}_{\text{cp},n}(f_n)}{\partial f_n}\right|_{f_n=\bar{f}_n} (f_n - \bar{f}_n) \triangleq \tilde{E}^{r-}_{\text{cp},n}. \quad (19)$$

After substituting $E^{r-}_{\text{cp},n}$ with $\tilde{E}^{r-}_{\text{cp},n}$ in (18), the objective function becomes:

$$(\mathbf{a}^\star, \mathbf{f}^\star) = \arg\min_{\mathbf{a},\mathbf{f}} \sum_{r\in R_k}\sum_{n\in\mathcal{F}} a^r_n \left(\underbrace{E^{r+}_{\text{cp},n} - \tilde{E}^{r-}_{\text{cp},n}}_{\tilde{E}^r_{\text{cp},n}} + E^r_{\text{comm},n}\right). \quad (20)$$

This transformed optimization problem is convex for fixed allocation variables, thus it can be solved by employing primal and dual decomposition methods [30], [34]. Primal decomposition can be applied when the problem has a coupling variable such that, when fixed to some value, the rest of the optimization problem decouples into several subproblems. Thus, let us decompose our objective problem to:

$$(\mathbf{a}^\star, \mathbf{f}^\star) = \arg\min_{\mathbf{a}} \arg\min_{\mathbf{f}} \sum_{r\in R_k}\sum_{n\in\mathcal{F}} a^r_n \left(\tilde{E}^r_{\text{cp},n} + E^r_{\text{comm},n}\right) \quad (21)$$

subject to (12) – (16). Now, according to (21), the solution to the optimization problem comes down to solving a two-step minimization problem. In the first step, the optimal CPU frequencies $\mathbf{f}^\star$ are determined for fixed allocation variables. First, let us define the auxiliary variables f^r_n determining the CPU frequencies of node n where request r is allocated. The relation between f^r_n and f_n is given by: $f_n = \sum_{r\in R_k} a^r_n f^r_n$ while satisfying constraints (12) and (13). The optimal values of allocation variables $\mathbf{a}^\star$ are obtained in the second step based on the previously determined $\mathbf{f}^\star$ and sets of constraints (12) – (13). Thus, we can now define the Lagrangian function of the subproblem for determining $\mathbf{f}^\star$:

$$\begin{aligned}\mathcal{L}(\mathbf{a}, \mathbf{f}, \boldsymbol{\mu}, \boldsymbol{\Phi}, \boldsymbol{\Psi}) &= \sum_{r\in R_k}\sum_{n\in\mathcal{F}} a^r_n \left(\tilde{E}^r_{\text{cp},n} + E^r_{\text{comm},n}\right) \\ &\quad - \sum_{n\in\mathcal{F}} \Phi_n (f_{\min,n} - f^r_n) - \sum_{n\in\mathcal{F}} \Psi_n (f^r_n - f_{\max,n}) \\ &\quad - \sum_{r\in R_k} \mu^r (D^r_{\text{tot}} - D^r_{\max}),\end{aligned} \quad (22)$$

and the Lagrange dual problem:

$$(\mathbf{a}^\star, \mathbf{f}^\star, \boldsymbol{\mu}^\star, \boldsymbol{\Phi}^\star, \boldsymbol{\Psi}^\star) = \arg\min_{\boldsymbol{\mu},\boldsymbol{\Phi},\boldsymbol{\Psi}\ge 0} \arg\min_{\mathbf{a}} \arg\min_{\mathbf{f}} \mathcal{L}(\mathbf{a}, \mathbf{f}, \boldsymbol{\mu}, \boldsymbol{\Phi}, \boldsymbol{\Psi}) \quad (23)$$

subject to (12), (13), where $\boldsymbol{\mu} = \{\mu^r\}, \forall r \in R_k, \mu^r \in \mathbb{R}^+$, $\boldsymbol{\Phi} = \{\Phi_n\}, \forall n \in \mathcal{F}, \Phi_n \in \mathbb{R}^+$ and $\boldsymbol{\Psi} = \{\Psi_n\}, \forall n \in \mathcal{F}$, $\Psi_n \in \mathbb{R}^+$ are the Lagrangian multipliers responsible for fulfilling sets of constraints (14) and (15), respectively. The dual problem (23) can be decomposed into a master problem and subproblems, and thus solved iteratively. Allocation variables $\mathbf{a}$ and CPU frequencies $\mathbf{f}$ are obtained by solving subproblems and then the Lagrange multipliers $\boldsymbol{\mu}, \boldsymbol{\Phi}, \boldsymbol{\Psi}$ are updated by solving the master problem for the obtained frequencies. This process continues until convergence while satisfying constraints.

A. Solving the Subproblems

The primal problem is solved in two steps. First, the optimal values of the CPU frequencies $\mathbf{f}^\star$ for each request $r \in R_k$ and node $n \in \mathcal{F}$ are obtained. Then, in the second step, the optimal values of the allocation variables $\mathbf{a}^\star$ are determined based on $\mathbf{f}^\star$. Thus, we can find the optimal CPU frequencies with Karush–Kuhn–Tucker (KKT) conditions for fixed allocation variables $\mathbf{a}$ by taking the partial derivative of (22) with respect to f_n^r and setting the gradient to 0:

$$\frac{\partial \mathcal{L}}{\partial f_n^r} = 0 \qquad \forall n \in \mathcal{F}, \forall r \in R_k. \tag{24}$$

Due to the polynomial form of the objective function and the set of constraints (14), there is no closed-form solution for the above equation. Therefore, a numerical method, e.g., the Newton method with the maximum number of iterations I_{num} has to be applied to solve it.

Vector $\mathbf{a}^\star$ can be obtained based on the optimal values of the CPU clock frequency determined in the first step by solving the following optimization problem.

$$\begin{aligned}\mathbf{a}^\star = \arg\max_{\mathbf{a}} & \sum_{r\in R_k}\sum_{n\in\mathcal{F}} a_n^r \left(\tilde{E}_{\text{cp},n}^{r}{}^\star + E_{\text{comm},n}^r\right) \\ & - \sum_{n\in\mathcal{F}} \Phi_n \left(f_{\min,n} - f_n^{r\star}\right) - \sum_{n\in\mathcal{F}} \Psi_n \left(f_n^{r\star} - f_{\max,n}\right) \\ & - \sum_{r\in R_k} \mu^r \left(D_{\text{tot}}^{r}{}^\star - D_{\max}^r\right),\end{aligned} \tag{25}$$

subject to (12), (13), where $\tilde{E}_{\text{cp},n}^{r}{}^\star = \tilde{E}_{\text{cp},n}^r\left(f_n^{r\star}\right)$ and $D_{\text{tot}}^{r}{}^\star = D_{\text{tot}}^r\left(f_n^{r\star}\right)$. The optimization problem defined in (25) is a linear assignment problem and can be solved by the Hungarian algorithm [35]. Let us define matrix $\boldsymbol{\Theta} = \left\{E_{\text{tot},n}^{r}{}^\star\right\}$, $\forall r \in R_k$ and $\forall n \in \mathcal{C}$ with $|R_k|$ rows and $|\mathcal{C}|$ columns, and matrix $\boldsymbol{\Lambda} = \left\{\tilde{E}_{\text{tot},n}^{r}{}^\star\right\}$, $\forall r \in R_k$ and $\forall n \in \mathcal{F}$ with the same number of rows and $|\mathcal{F}|$ columns, where $\tilde{E}_{\text{tot},n}^{r}{}^\star = \tilde{E}_{\text{cp},n}^{r}{}^\star + E_{\text{comm},n}^r$. To reflect unlimited computational resources at each CN, we introduce matrix $\boldsymbol{\Omega} = \left[\boldsymbol{\Lambda}\ \boldsymbol{\Theta} \otimes \mathbf{1}_{1\times|R_k|}\right]$, where $\otimes$ is the Kronecker tensor product, while $\mathbf{1}_{1\times|R_k|}$ is a vector of ones with one row and $|R_k|$ columns. It means that the columns of $\boldsymbol{\Theta}$ are replicated $|R_k|$ times and matrix $\boldsymbol{\Omega}$ has $|R_k|$ rows and $|R_k| \cdot |\mathcal{C}| + |\mathcal{F}|$ columns. E.g., , in the case of three tasks $|R_k| = 3$, two fog nodes $\mathcal{F} = \{1, 2\}$ and one cloud $\mathcal{C} = \{3\}$, matrix $\boldsymbol{\Omega}$ is defined as follows:

$$\boldsymbol{\Omega} = \begin{bmatrix} \underbrace{\begin{matrix}\tilde{E}_{\text{tot},1}^{1}{}^\star & \tilde{E}_{\text{tot},2}^{1}{}^\star \\ \tilde{E}_{\text{tot},1}^{2}{}^\star & \tilde{E}_{\text{tot},2}^{2}{}^\star \\ \tilde{E}_{\text{tot},1}^{3}{}^\star & \tilde{E}_{\text{tot},2}^{3}{}^\star\end{matrix}}_{E_{\text{tot},n}^{r}{}^\star\ \forall n\in\mathcal{F}} & \underbrace{\begin{matrix}E_{\text{tot},3}^{1}{}^\star & E_{\text{tot},3}^{1}{}^\star & E_{\text{tot},3}^{1}{}^\star \\ E_{\text{tot},3}^{2}{}^\star & E_{\text{tot},3}^{2}{}^\star & E_{\text{tot},3}^{2}{}^\star \\ E_{\text{tot},3}^{3}{}^\star & E_{\text{tot},3}^{3}{}^\star & E_{\text{tot},3}^{3}{}^\star\end{matrix}}_{E_{\text{tot},n}^{r}{}^\star\ \forall n\in\mathcal{C}} \end{bmatrix} \tag{26}$$

Next, applying the Hungarian algorithm for matrix $\boldsymbol{\Omega}$, the matrix with binary values is determined, e.g., :

$$\mathcal{H}(\boldsymbol{\Omega}) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{27}$$

The example above shows that the first task is computed in the second fog node while the second and the third tasks are computed in the CN. Thus, the optimal values of $a_n^{r\star}$ can be determined by:

$$a_n^{r\star} = \begin{cases} \mathcal{H}(\boldsymbol{\Omega}(r,n)) & \text{for } n \le |F| \\ \sum_{j=|F|+1}^{|\mathcal{F}|+|R_k||\mathcal{C}|+1} \mathcal{H}(\boldsymbol{\Omega}(r,j)) & \text{for } n > |F|. \end{cases} \tag{28}$$

B. Solving the Master Problem

We can fulfill sets of constraints (14) and (15) by determining the search range of the optimal solution. Let $f_{\text{delay},n}^r$ denote the minimum value of f_n which satisfies constraint (14) for request $r \in R_k$ processed at node $n \in \mathcal{F}$. These can be obtained by solving the following equation:

$$D_{\text{tot},n}^r - D_{\max}^r = 0, \quad \forall n \in \mathcal{F}, \forall r \in R_k. \tag{29}$$

Inserting $D_{\text{tot},n}^r$ from Eq. (9) (together with $D_{\text{cp},n}^r$ taken from Eq. (8)) into Eq. (29) we get:

$$f_{\text{delay},n}^r = \frac{L^r\theta^r}{s_n\left(D_{\max}^r - \tau_{\text{req}}^r - D_{\text{comm},n}^r - D_{\text{queue},n}^r - \tau_{\text{res}}^r\right)}. \tag{30}$$

If $f_{\text{delay},n}^r > f_{\max,n}$, task r cannot be processed in FN n within the delay constraint. If $f_{\text{delay},n}^r \le f_{\min,n}$, the minimum CPU clock frequency is kept at $f_{\min,n}$. Thus, if the obtained optimal clock frequency is in the range $f_n \in \langle \max\left\{f_{\min,n}, f_{\text{delay},n}^r\right\}, f_{\max,n}\rangle$, the Lagrange multipliers in (22) and (25) are simplified by setting $\Phi_n = 0$, $\Psi_n = 0$, $\forall n \in \mathcal{F}$ and $\mu^r = 0$, $\forall r \in R_k$.

Finally, we propose an algorithm called EEFFRA (Alg. 1) to finding the solution (CPU clock frequencies and request allocation over the nodes) for total energy minimization with delay constraints, as discussed above. The computational complexity of the proposed algorithm results from the complexity of the Hungarian algorithm (line 8) and the iterative frequency finding procedure (lines 3-7). The complexity of the Hungarian algorithm is proportional to a cube of a greater number: the number of tasks or the number of agents. In our model, the requests (tasks) can be assigned to the fog nodes or the cloud (agents). Moreover, the cloud can process more than one request simultaneously. Therefore, in our model, we have $|R_k|$ tasks that can be assigned to $|\mathcal{F}| + |R_k||\mathcal{C}|$ nodes, where $|R_k||\mathcal{C}|$ represents the cloud nodes that can process more than one request.

Thus, in our solution the complexity of the Hungarian algorithm equals $\mathcal{O}\left((|\mathcal{F}| + |R_k||\mathcal{C}|)^3\right)$. The second part of the computational complexity of EEFFRA results from CPU frequency calculation. Let us observe that CPU frequency has to be calculated for each node and each request i.e., $|R_k|\,|\mathcal{N}|$ frequencies have to be determined. The main step of the proposed algorithm (line 5) determines the optimal CPU frequencies for a given approximation of the objective

Algorithm 1 EEFFRA in fog computing networks.

```
1: Inputs: $L^r$, $\theta^r$, $o^r$, $D^r_{\text{max}}$ for $r \in R_k$, $\{p_{n,0}, p_{n,3}, \dots, p_{n,Q}\}$, $f_{\text{min},n}$, $f_{\text{max},n}$, $s_n$, $d^n$ for $n \in \mathcal{F}$, $\gamma^r_n$, $b^n_r$ for $r \in R_k$ and $n \in \mathcal{F}$ and $b_{\text{back}}$, $\chi$, maximum number of iterations $I_{\text{num}}$, $I_{\text{sca}}$, iteration index $i_{\text{sca}}$, maximum error $\varepsilon$ and initial values of optimization variables $\mathbf{f}^\star$
2: Outputs: $E^r_{\text{tot},n}$ for $r \in R_k$ and $n \in \mathcal{F}$, $\mathbf{f}^\star$, $\mathbf{a}^\star$
3: repeat
4:     $\bar{\mathbf{f}} \leftarrow \mathbf{f}^\star$
5:     calculate $\mathbf{f}^\star$ by solving (24) in the range $f_n \in \left\langle \max\left\{f_{\text{min},n}, f^r_{\text{delay},n}\right\}, f_{\text{max},n}\right\rangle$ for $\Phi_n = 0$, $\Psi_n = 0$, $\forall n \in \mathcal{F}$ and $\mu^r = 0$, $\forall r \in R_k$ using numerical method with max. $I_{\text{num}}$ iterations
6:     $i_{\text{sca}} \leftarrow i_{\text{sca}} + 1$
7: until $\left|\mathbf{f}^\star - \bar{\mathbf{f}}\right| \leq \varepsilon$ or $i_{\text{sca}} = I_{\text{sca}}$
8: calculate $\mathbf{a}^\star$ using the Hungarian method for matrix $\boldsymbol{\Omega}$ and (28)
9: $E^r_{\text{tot},n} \leftarrow \tilde{E}^r_{\text{tot},n}$ for $r \in R_k$ and $n \in \mathcal{F}$
```

function which are then updated in the loop (line 5). This procedure is repeated until the termination conditions are met (line 7). Thus, the complexity of the CPU frequency calculation is equal to $\mathcal{O}\left(|R_k|\,|\mathcal{N}|\,i_{\text{num}} i_{\text{sca}}\right)$, where i_{num} and i_{sca} are the numbers of iterations of the numerical method applied to solve (24) and the SCA method, respectively. The complexity of the entire algorithm is therefore equal to $\mathcal{O}\left(\left(|\mathcal{F}| + |R_k||\mathcal{C}|\right)^3 + |R_k|\,|\mathcal{N}|\,i_{\text{num}} i_{\text{sca}}\right)$.

C. Low-Complexity EEFFRA (LC-EEFFRA)

We remove the Hungarian algorithm from EEFFRA in the following approach to our optimization problem leading to reduced computational complexity $\mathcal{O}\left(|R_k|\,|\mathcal{N}|\,i_{\text{num}} i_{\text{sca}}\right)$. The optimal values of frequency $\mathbf{f}^\star$ are determined in the same way as in Alg. 1, while the values of $\mathbf{a}^\star$ are obtained in a heuristic manner. In this heuristic approach, only a single request $r \in R_k$ is considered at a time. It is allocated to node $n^\star$ where the energy consumption for processing r is the lowest, i.e., we find:

$$n^\star = \arg\min_n \tilde{E}^r_{\text{tot},n}{}^\star \; \forall r \in R_k. \tag{31}$$

Values $t_{n,k}$ are updated after the allocation of each request to prevent multiple collisions of two or more requests at the same FN. The examination order of requests arriving at the same time is random to emulate a decentralized approach.

VI. Results

Results obtained from computer simulations are presented in this section. Parameterization and simulation setup are described in Section VI-A. Section VI-B shows a discussion on convergence and optimality of EEFFRA. Simulation results are presented in Sections VI-C through VI-F.

TABLE II
Simulation Parameters

Symbol	Value/Range	Symbol	Value/Range
Requests, $r \in R_k$			
L^r	[1,10] MB	θ^r	[1,100] FLOP/bit
o^r	[0, 0.5]	D^r_{max}	[100, 1000] ms
$\|R_k\|$	[5,10]	$\overline{T_k - T_{k-1}}$	40 ms
Number of nodes			
$\|\mathcal{F}\|$	10	$\|\mathcal{C}\|$	1
Computations in the fog [26], [27], [36], $n \in \mathcal{F}$			
s_n	16 FLOP/cycle	Q	3
$p_{n,3}$, $p_{n,2}$	5.222, 34.256	$p_{n,1}$, $p_{n,0}$	88.594, -47.152
$f_{\text{min},n}$	1.6 GHz	$f_{\text{max},n}$	4.2 GHz
Computations in the cloud [25], [26], $n \in \mathcal{C}$			
f_n	1.5 GHz	s_n	32 FLOP/cycle
Fog-to-fog and fog-to-cloud transmission [28], [43], [44]			
d^n, $n \in \mathcal{C}$	2000 km	χ	7.5 µs/km
b_{back}	1 Gbps	b^n_r, $n \in \mathcal{F}$	1 Gbps
γ^r_n, $n \in \mathcal{F}$	0.3 nJ/(bit·hop)	γ^r_n, $n \in \mathcal{C}$	10 nJ/bit
Device-to-fog transmission [38], [39], [41]			
Modulation	64-QAM	Spatial Streams	2
Coding rate	2/3	Channel bandwidth	80 MHz
Users per AP	5	N_{NDBPS}	1872
Frame payload	2000 B	Frames per A-MPDU	25

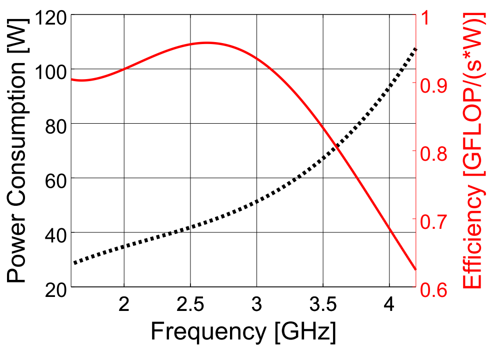


Fig. 2. Power consumption and energy efficiency of Intel Core i5-2500K vs. CPU frequency.

A. Parameterization and Simulation Setup

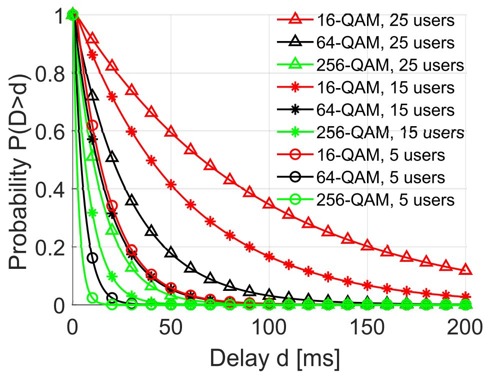
Let us consider a network with $|\mathcal{F}| = 10$ FNs and $|\mathcal{C}| = 1$ cloud DC. Simulation parameters are summarized in Table II. The process of generating requests for simulations is as follows. At time $T_k \in \mathcal{T}$ there appear between 5 and 10 (uniform distribution) new computational requests. Value T_k is generated at a random delay after previous time instance T_{k-1}. The difference $T_k - T_{k-1}$ is chosen to be a random variable of exponential distribution with an average value of 40 ms (intensity 20 s^{-1}). The requests have randomly (with uniform distribution) assigned values of their parameters (size, arithmetic intensity, delay requirement) in ranges shown in Table II.

It is assumed that each FN uses a single Intel Core i5-2500K as its CPU. Data relating the frequency, voltage, and power consumption of i5-2500K is taken from [36] and fit into Eq. (2) adapted from [27]. The resulting power consumption and energy efficiency are plotted in Fig. 2. These figures show that power consumption increases faster-than-linearly with operating frequency and that the frequency with the highest energy efficiency is at 2.6063 GHz. The cloud CPUs are parameterized according to the *Intel Xeon Phi* family commonly used in computer clusters [25], [37] characterized by $s = 32$ FLOP/cycle [26], and run at a constant frequency of 1.5 GHz. Fog-to-fog and fog-to-cloud transmission parameters are analogous to those used in [21].

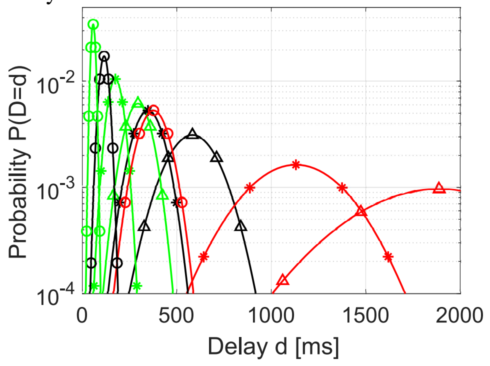
We assume IEEE 802.11ac [38] for parameterizing End User-to-Access Point delay. In our scenario, requests are wirelessly transmitted to FNs co-located with Access Points (APs). We assume that the transmission uses the Request To Send/Clear To Send (RTS/CTS) mechanism and Aggregate Medium Access Control (MAC) Protocol Data Unit (A-MPDU) frames [38], [39]. In such a case, the MAC delay is modeled as a quasi-exponential distribution according to the procedure shown in [40]. We modify it by including PHY layer delay [38], [41] and by parameterizing it with values characterizing 802.11ac [38], [39]. The model takes the number of devices connected to a given AP as an input parameter since these devices compete with each other for a channel and can potentially interfere with each other increasing the transmission delay. Moreover, the transmission rate depends on the chosen Modulation Coding Scheme (MCS), the number of spatial streams, and the channel bandwidth. These parameters directly impact the N_{NDBPS} parameter of transmission – the number of useful (i.e., not used on control and coding) data bits per symbol. A higher N_{NDBPS} translates to a higher data rate. On the other hand, the ability to adjust these parameters can be limited. E.g., not every device supports Multiple-Input Multiple-Output (MIMO) transmission required to transmit multiple spatial streams and the chosen MCS must reflect the quality of the link – only lower MCSs can be used when the quality of the link is poor. The quality of the link (which can be expressed using, e.g., Signal-to-Noise Ratio (SNR)) can be influenced by a multitude of factors. The key component is the path loss between the transmitter and the receiver which can vary in time due to phenomena such as shadowing (large-scale fading caused by obstacles) or multipath fading. Other components include the Doppler shift, in-band and out-of-band interference. In practice, MCS selection algorithms switch between MCSs depending on the number of lost frames/packets [42]. An example of empirical results examining the selection of MCSs based on SNR is shown in [42].

Let us show the delay profile of transmission derived from these models in Figs. 3a and 3b. The results are plotted for various MCSs, numbers of spatial streams, and numbers of users communicating with the same AP (these numbers represent the total number of users utilizing a given AP, not just those offloading computational tasks, as the same AP can be used by different users for various reasons e.g., browsing the web, streaming, uploading and downloading files). A long (800 ns) guard interval is used in all examples as the support for a short (400 ns) guard interval is only optional for Very High Throughput (VHT) MCSs.

Fig. 3a shows the complementary Cumulative Distribution Function (CDF) of the delay for a single transmitted A-MPDU frame aggregating 25 MAC Protocol Data Units (MPDUs). The transmission delay of a single frame follows a quasi-exponential (slightly shifted and scaled) distribution. The rate depends both on the number of users and the N_{NDBPS} parameter of transmission. Fig. 3b shows the Probability Density



(a) Single A-MPDU frame (25×2000 B payload) delay



(b) Request (1 MB – 20 A-MPDU frames) delay

Fig. 3. Wireless transmission delay for an 80 MHz channel. Examined MCS are: 16-QAM with coding rate R=1/3 and 1 spatial stream, $N_{\mathrm{NDBPS}} = 468$; 64-QAM with R=2/3 and 2 spatial streams, $N_{\mathrm{NDBPS}} = 1872$; 256-QAM with R=3/4 and 4 spatial streams, $N_{\mathrm{NDBPS}} = 5616$.

Function (PDF) of the delay of a 1 MB request calculated as a sum of 20 A-MPDU delays, each A-MPDU aggregating 25 MPDUs. These probabilities (in accordance with the Central Limit Theorem) resemble normal distributions. Both plots show that the delay increases significantly with an increasing number of users sharing a channel as a result of contention and potential collisions. More efficient MCS reduces both the mean value and the variance of the transmission delay.

If not stated differently, simulations for each data point are obtained over 5050 time instances T_k. Results from the first 50 instances are discarded. Random number generator seeds are kept the same for each value of swept parameters for a fair comparison of results.

Our solutions, i.e., EEFFRA and LC-EEFFRA, are compared with three reference methods. The first method, called *Cloud Only*, processes all requests in the cloud. The second method, called *Fog Only*, processes all requests in the FNs. The FNs' CPU frequencies and requests-to-nodes assignments are determined with LC-EEFFRA. Finally, the third method, called *Fog Simple*, processes requests in the same FN that these requests arrived at. Still, it uses optimal FNs' CPU frequencies determined using LC-EEFFRA. Simulations are performed using MATLAB.

B. Convergence of Algorithms and Optimality of Solution

EEFFRA utilizes SCA for finding optimal operating frequencies and the Hungarian algorithm for assigning requests

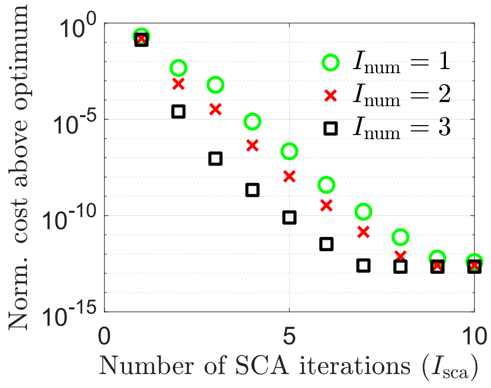


Fig. 4. Convergence of solutions found by EEFFRA to optimum with the number of iterations.

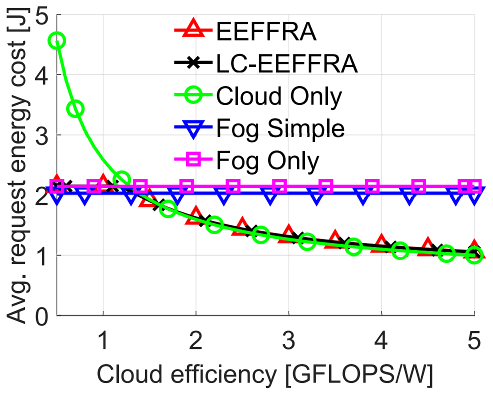


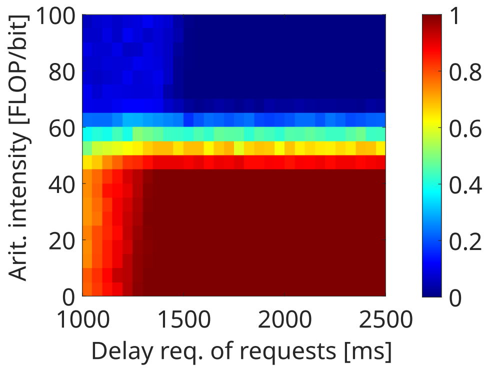
Fig. 5. Influence of cloud energy efficiency on average energy cost for chosen policies.

to nodes. The Hungarian algorithm is guaranteed to find the optimum in polynomial time [35]. SCA is guaranteed to converge [45]. To show the SCA convergence rate, we plot normalized energy costs resulting from offloading requests depending on the maximum number of algorithm iterations in Fig. 4. Apart from SCA iterations I_{sca}, we also vary the maximum number of iterations I_{num} used to find the optimum CPU frequencies in Alg. 1. For clarity of convergence analysis, we assume that there is no cloud, i.e., $|\mathcal{C}| = 0$. The operating frequency of the cloud is not adjusted (not a variable) and therefore does not influence the analysis. Normalization is obtained by plotting the relative difference between the cost achieved by EEFFRA and the optimal cost found by solving the original problem without SCA. Since the degree of the polynomial which models CPU power consumption in these simulations equals 3, the optimal frequencies $f_n^{r\star}$ can be found analytically for each request $r \in R_k$ and node $n \in \mathcal{F}$ which result in the lowest cost $E^r_{\text{cp},n}$ while fulfilling (14). Since the first derivative of $E^r_{\text{cp},n}$ over f_n from (17) is continuous everywhere except at singularity at $f_n = 0$, and has at most 3 real roots, the optimal frequency $f_n \in \langle \max\left\{f_{\min,n}, f^r_{\text{delay},n}\right\}, f_{\max,n}\rangle$ is either obtained for the endpoint of this interval or for one of the roots of $\frac{d}{df_n}E^r_{\text{cp},n}(f_n)$. The lowest value $E^r_{\text{cp},n}$ for these frequencies determines the optimal frequency $f_n^{r\star}$. The solution is continued as described in Section V-A from Eq. (25) onwards – the linear assignment problem is solved using the Hungarian algorithm. It is visible in Fig. 4 that increasing the maximum number of iterations I_{sca} and I_{num} moves EEFFRA in the direction of the optimal solution. EEFFRA converges both quickly and to values close to the optimum (the relative difference to the optimum is lower than 10^{-7} with $I_{\text{num}} = 3$ after just 3 SCA iterations, and it drops below 10^{-12} after 7 iterations). Meanwhile, IEEE 754 double-precision numbers (which our simulations are based on) have a precision of $2^{-53} \approx 10^{-16}$.

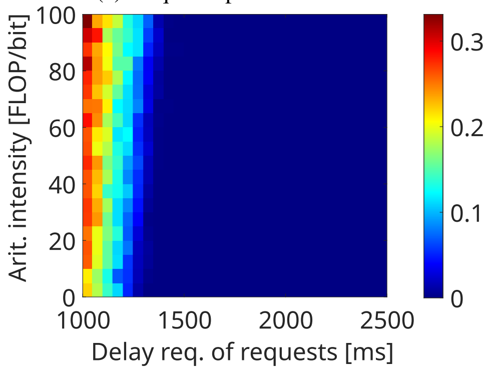
C. Impact of Computational Energy Efficiency of the Cloud

First, we vary the values of computational efficiency of the cloud DC in the range [0.5, 5.0] GFLOP/(s·W) (the median value for 500 of the most powerful commercially available computer clusters equals 2.962 GFLOP/(s·W) [25]). The average energy costs per successfully processed request are shown in Fig. 5. At low computational efficiency of the cloud, policies utilizing only nodes in the fog tier (*Fog Simple*, *Fog Only*) perform similarly to those utilizing both the fog and the cloud. The *Cloud Only* approach has the highest energy consumption at low efficiency of the cloud (up to around 1.3 GFLOP/(s·W)). This threshold is relatively low (compared with efficiency values found in [25]) as the examined CPUs working in FNs are not the most efficient ones as for 2022. Above that level *Cloud Only* is characterized by lower energy consumption than *Fog Simple* and *Fog Only* and similar to EEFFRA and LC-EEFFRA as under these parameters it is the most efficient to process most requests in the cloud. EEFFRA is slightly more efficient than LC-EEFFRA at higher cloud efficiency values. The percentage of requests which were unable to be processed using each of the offloading policies is the following: The *Fog Simple* solution where FNs cannot "share" computational requests between themselves has the highest ratio of rejected requests (8.1%), while 3.7%-4.5% requests (percentage varies depending on cloud efficiency – lower for higher efficiency) are rejected by both proposed solutions utilizing both fog and cloud (EEFFRA and LC-EEFFRA). *Fog Only* and *Cloud Only* have rejection rates of 4.5% and 6.5% respectively. Requests which are rejected tend to have larger sizes and higher arithmetic intensities (as shown later in Figs 6b and 9). It "artificially" decreases the average-per-request cost of methods with higher rejection rates in Fig. 5, e.g., causing *Fog Simple* to show a lower average cost than *Fog Only*.

Let us examine more closely where EEFFRA chooses to offload computational requests and what parameters impact these decisions. Fig. 6 shows histograms of parameters characterizing offloaded requests obtained after running simulations for 50000 T_k instances. Fig. 6a and Fig. 6b show the probabilities of requests being processed in the fog tier of the network and those rejected due to too low delay requirement at cloud efficiency 1.3 GFLOP/(s·W). A similar histogram for requests processed in the cloud would be superfluous, as the probabilities for results processed in the fog, in the cloud, and those rejected sum up to 1. Unsurprisingly, Fig. 6b shows



(a) Requests processed in FNs



(b) Rejected requests.

Fig. 6. Histograms of requests at 1.3 GFLOP/(s·W) cloud efficiency. Results of EEFFRA.

that results with strict latency requirements are less likely to be successfully processed in time. In Fig. 6a one can see a "threshold" between 45 and 50 FLOP/bit below which all requests are chosen by EEFFRA to be served by the FNs rather than the cloud. Similar histograms plotted for other values of cloud efficiencies show that this threshold increases with a less efficient cloud and decreases with a more efficient cloud. In particular, for efficiencies below 1.0 GFLOP/(s·W), all requests are processed in the fog. These thresholds are "blurry" – the rate of requests processed in the fog tier does not jump to 0 for requests above 50 FLOP/bit in Fig. 6a. For requests in the range from 45 to 65 FLOP/bit, the optimal offloading decisions depend on other factors, such as the state of the network (potential queues in the FNs), and the size of the requests. On the other hand, even for infinitely high efficiencies of the cloud, around 20% of tasks remain processed in the fog tier (low-intensity ones for which the cost of transmission to the cloud outweighs computational costs in the fog and those with both large size and strict delay requirements). In Fig. 6a up to 11% of tasks with arithmetical intensity above 70 FLOPS/bit and with delay requirements below 1550 ms are processed in the fog tier as a result of the cloud being unable to fulfill these requirements.

To better illustrate the differences in request allocation policies, we plot the CDF of energy cost spent on a single request. Energy spent on rejected requests is assumed to be infinite for the purpose of CDF plots. Such results can be seen in Fig. 7. First, it is visible that the utilization of both fog and cloud tiers of the network yields significantly better results than utilizing nodes in only one tier. The proposed EEFFRA and LC-EEFFRA provide the lowest required energy cost for each percentile of the CDFs. All methods do not reach 1 on the y-axis, i.e., some requests cannot be processed within a given maximum tolerated delay. Our methods achieve the lowest rejection rate as shown in the inset of Fig. 7.

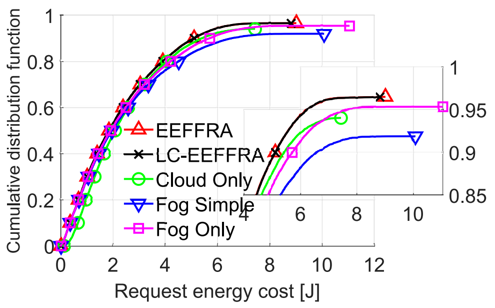


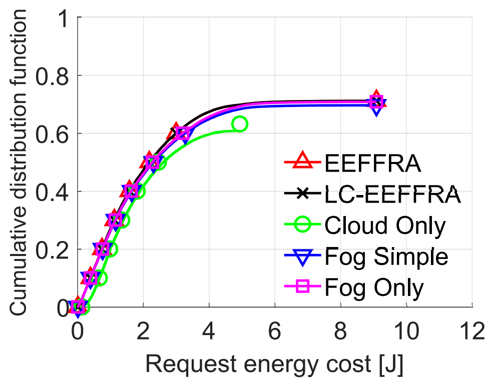
Fig. 7. CDFs of request processing energy cost at cloud efficiency of 1.3 GFLOP/(s·W). Comparison of different policies.

D. Impact of Delay Requirements and Size of Requests on the Offloading Decisions

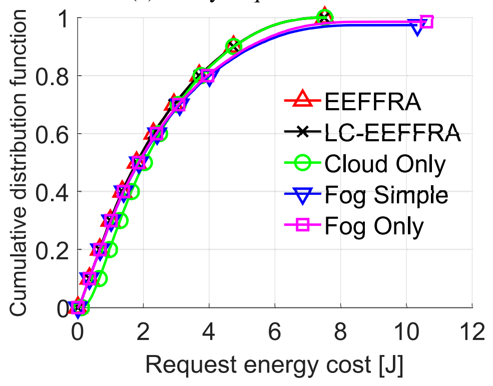
Let us see how the energy consumption and the percentage of rejected requests change with the size and the delay requirement of computation requests. The energy efficiency of the cloud is set to 1.3 GFLOP/(s·W) and parameter sweeps for other parameters are performed. All other parameters are taken from Table II.

Fig. 8 plots CDFs of energy consumption costs of processing single requests with delay requirements: 1000 ms (Fig. 8a) and 2500 ms (Fig. 8b). At the required delay of 1000 ms, all methods have high rejection rates, with *Cloud Only* being clearly the worst-suited for low-latency applications. The differences between the rest of the methods are minor – the requests with such low delay requirements either can or cannot be solved in time at the receiving FN and the ability of nodes to transmit tasks between themselves does not significantly improve performance. With a required delay of 2500 ms, the differences between approaches become more profound. Utilizing both fog and cloud (EEFFRA, LC-EEFFRA) gives the lowest rejection rates and energy costs. While *Cloud Only* achieves similar energy costs for requests above the 90th percentile, it has the worst performance below the 60th percentile. It stems from the fact that for requests with low arithmetic intensity, the high efficiency of the cloud is not enough to offset the costs caused by long-range transmission.

Fig. 9 plots CDFs of energy consumption costs of processing single requests with sizes 5 MB, (Fig. 9a) and 10 MB (Fig. 9b). Fig. 9 shows that EEFFRA and LC-EEFFRA achieve the lowest energy costs and rejection rates for both 5 MB and 10 MB requests. For 10 MB requests, rejection rates achieved by EEFFRA are 21 and 22.8 percentage points lower than those of *Fog Only* and *Fog Simple* respectively. LC-EEFFRA achieves the same performance as EEFFRA. It is worth observing that for both request sizes *Cloud Only* has the highest energy costs up to at least the 20th percentile (caused

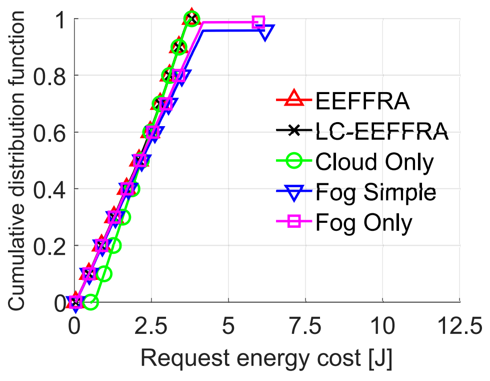


(a) Delay req.: 1000 ms

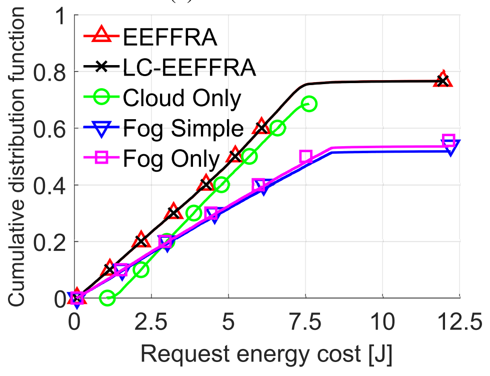


(b) Delay req.: 2500 ms

Fig. 8. CDFs of request processing energy cost – influence of delay requirement of requests.

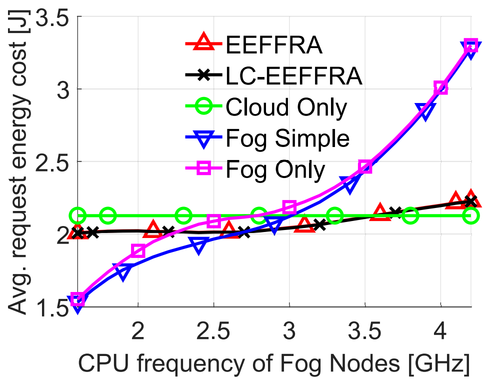


(a) Size: 5 MB.

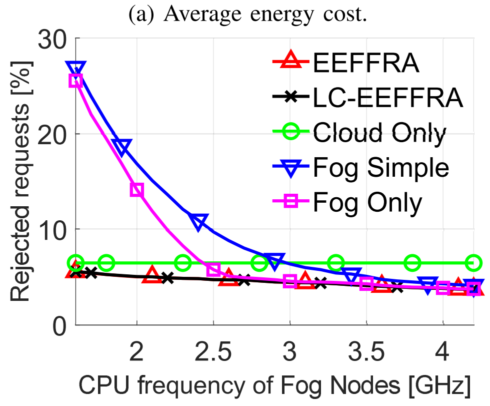


(b) Size: 10 MB.

Fig. 9. CDFs of request processing energy cost – influence of request size.



(a) Average energy cost.



(b) Rejected requests.

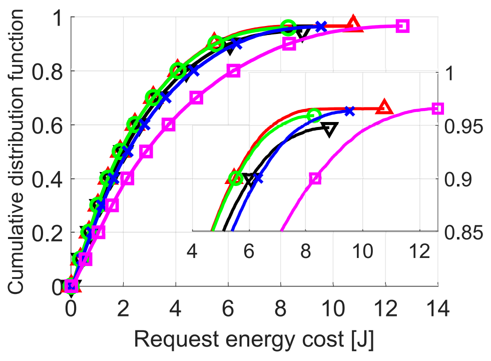
Fig. 10. Influence of fixed CPU frequency of FNs.

by energy spent for transmission), but for higher percentiles (influenced by requests with higher arithmetical intensities) its costs are lower than that of either *Fog Only* or *Fog Simple*.

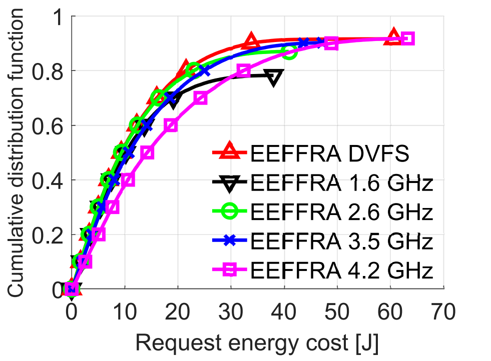
E. Impact of CPU Frequency of Fog Nodes

In the previous sections, it is assumed that FNs can dynamically adjust their operating frequency (and voltage) to minimize energy consumption while satisfying delay requirements. Let us assume that all FNs utilize the same, fixed CPU frequency. Fig. 10 shows the average energy cost and percentage of rejected requests plotted as a function of this fixed frequency (swept between 1.6 and 4.2 GHz with a 0.1 GHz step). The results for *Cloud Only* are constant, as no requests are processed in FNs. *Fog Simple* and *Fog Only* methods have high rejection rates at low frequencies. Meanwhile, EEFFRA and LC-EEFFRA have the lowest rejection rates while also having the lowest (considering the rejected requests are not taken into account by this metric) energy costs. As the frequency of FNs increases, the rejection rates decline and average energy cost increases for all methods utilizing FNs. However, this effect is considerably weaker for EEFFRA and LC-EEFFRA (utilizing resources in both fog and cloud tiers) than for *Fog Simple* and *Fog Only*.

Let us compare the efficiency of the network employing EEFFRA with and without DVFS. As shown in Fig. 10 the possibility to send requests to the cloud diminishes the impact of FNs' operating frequency on energy costs and rejection rate. Therefore, to focus on the differences, Fig. 11 shows the results of simulations for a network with 10 FNs and no



(a) Intensity: [1, 100] FLOP/bit.



(b) Intensity: [1, 500] FLOP/bit.

Fig. 11. CDFs of request processing energy cost – comparison of FNs working at fixed frequencies and utilizing DVFS.

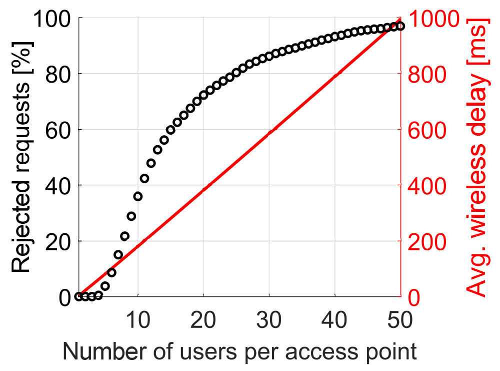
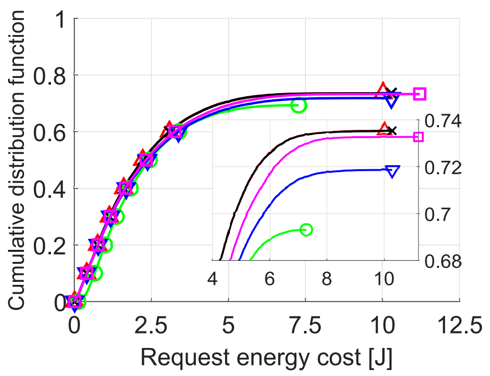
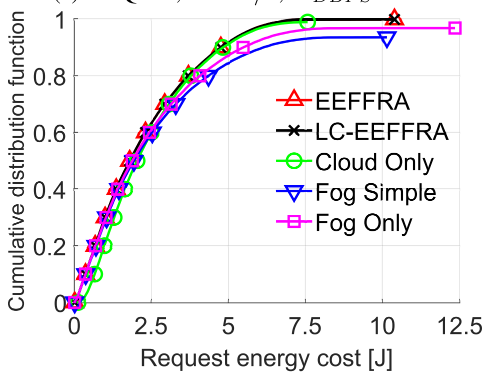


Fig. 12. Influence of number of users on wireless transmission delay and percentage of rejected requests (EEFFRA).



(a) 16-QAM, $R = 1/2$, $N_{DBPS} = 936$



(b) 256-QAM, $R{=}3/4$, $N_{DBPS}{=}2808$

Fig. 13. CDFs of request processing energy cost – influence of varied wireless transmission parameters – modulation and coding.

connection to the cloud. We compare CDFs of energy costs per request achieved utilizing DVFS with the following fixed frequencies of FNs: 1.6 GHz (minimal), 2.6063 GHz (optimal frequency for maximizing energy efficiency as seen in Fig. 2, later referred to as 2.6 GHz), 4.2 GHz (maximal), and 3.5 GHz (halfway between optimal and maximal). The mean time between sets of requests is increased ($\overline{T_k - T_{k-1}} = 500$ ms), while in Fig. 11b, the range of possible arithmetic intensities of requests is increased to [1, 500] to make the requests highly variable in terms of required computations speed. Fig. 11b shows both significantly higher costs of processing requests and clearer differences between results generated for chosen frequencies. In both plots, rejection rates are increasing with decreasing fixed FN frequency. On the other hand, 4.2 GHz has the highest energy cost. EEFFRA utilizing DVFS manages to maintain the lowest energy cost for every percentile. In Fig. 11a the 75th percentile of EEFFRA cost is lower than that of 1.6 GHz, 2.6 GHz, 3.5 GHz, and 4.2 GHz by 6.8%, 0.3%, 13.2%, and 34.4%, respectively. In Fig. 11b these values change to: 24.7%, 3.6%, 13.4%, and 33.7%, respectively.

F. Impact of Wireless Access Delay

Let us examine how changing wireless transmission parameters impacts the processing of requests. More users communicating in Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA) system increases the likelihood of collisions, resulting in increased transmission delays. This impact can be seen in Fig. 12. The red line shows how the average wireless transmission delay of a request changes with the increasing number of users communicating with AP. It grows slightly faster than linearly. Black circles represent the percentage of requests which fail to meet their delay requirements under EEFFRA policy. The value for 5 users (3.8%) corresponds to the results shown in Fig. 7. The rejection rate increases rapidly with each additional user after 5 and breaks 50% for 13 users.

As shown earlier in Fig. 3, the used MCS has an impact on transmission delays. Fig. 13 shows CDFs of offloaded requests energy costs for two MCSs: 16-QAM with the coding rate of 1/2, and 256-QAM with 3/4 rate. These results can be directly compared to Fig. 7 where 64-QAM with 2/3 rate is used. In

Fig. 13a rejection rates increase for all examined policies. While differences between the results of policies become less profound, EEFFRA still outperforms other schemes. In Fig. 13b rejection rates are lower – the policies are able to successfully offload large requests with high arithmetic intensity. Interestingly, *Cloud Only* has lower rejection rates than both *Fog Simple* and *Fog Only* in Fig. 13b, and higher in Fig. 13a. Altogether, the shapes of CDFs in Fig. 13 correspond to those seen in Fig. 8. An increase in wireless transmission delay causes stricter delay requirements for wired transmission and computations. From Fig. 8a to Fig. 8b the level of tolerated delay is increased directly. Meanwhile, increasing rate of transmission from Fig. 13a to Fig. 13b indirectly has the same impact.

VII. Conclusion

We have formulated the optimization problem of minimizing the energy consumption in the fog computing network while maintaining the latency constraints. This energy consumption is assumed to result from both transmission and processing of offloaded computational tasks (computation requests originating from end-users). The latency and energy consumption models and their parameters are based on real-life computing and networking equipment product data sheets, and measurements.

The proposed EEFFRA algorithm solves the proposed optimization problem using its successive approximations for adjusting clock frequencies of CPUs in fog nodes. A suboptimal, lower complexity solution LC-EEFFRA, which does not require coordinated decision making, is also examined. Our algorithms allow for the flexibility needed to direct a task to the most favorable node and can lower request rejection rates by over 20 percentage points for large (10 MB) requests compared with baseline solutions. Additionally, utilizing DVFS can noticeably decrease computation-related energy consumption (by over 33% when compared with computing at the maximal frequency) while fulfilling delay requirements, even compared to fog nodes working at optimal but fixed frequencies (by 3.6% for computationally intensive requests). The proposed algorithms can be seen as promising solutions for managing fog computing networks.

References

[1] P. Cerwall et al., "Ericsson mobility report November 2021," Ericsson, Stockholm, Sweden, Tech. Rep. EAB-21:010887, 2021.

[2] Cisco, "Cisco global cloud index: Forecast and methodology, 2016–2021," Cisco, San Jose, CA, USA, White Paper C11-738085-02, 2018.

[3] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st, Ed., MCC workshop Mobile Cloud Comput. (MCC)*, 2012, pp. 13–16.

[4] D. Chen, S. Schedler, and V. Kuehn, "Backhaul traffic balancing and dynamic content-centric clustering for the downlink of fog radio access network," in *Proc. IEEE 17th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jul. 2016, pp. 1–6.

[5] J. Xu, K. Ota, and M. Dong, "Saving energy on the edge: In-memory caching for multi-tier heterogeneous networks," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 102–107, May 2018.

[6] K. Wang, J. Li, Y. Yang, W. Chen, and L. Hanzo, "Energy-efficient multi-tier caching and node association in heterogeneous fog networks," in *Proc. IEEE 92nd Veh. Technol. Conf. (VTC-Fall)*, Nov. 2020, pp. 1–5.

[7] Z. Liu, Y. Yang, K. Wang, Z. Shao, and J. Zhang, "POST: Parallel offloading of splittable tasks in heterogeneous fog networks," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3170–3183, Apr. 2020.

[8] M. K. Hussein and M. H. Mousa, "Efficient task offloading for IoT-based applications in fog computing using ant colony optimization," *IEEE Access*, vol. 8, pp. 37191–37201, 2020.

[9] L.-A. Phan, D.-T. Nguyen, M. Lee, D.-H. Park, and T. Kim, "Dynamic fog-to-fog offloading in SDN-based fog computing systems," *Future Gener. Comput. Syst.*, vol. 117, pp. 486–497, Apr. 2021.

[10] R. Lin et al., "Distributed optimization for computation offloading in edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 12, pp. 8179–8194, Dec. 2020.

[11] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017.

[12] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2016.

[13] L. Liu, Z. Chang, and X. Guo, "Socially aware dynamic computation offloading scheme for fog computing system with energy harvesting devices," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1869–1879, Jun. 2018.

[14] S. Sarkar and S. Misra, "Theoretical modelling of fog computing: A green computing paradigm to support IoT applications," *IET Netw.*, vol. 5, no. 2, pp. 23–29, 2016.

[15] S. Sarkar, S. Chatterjee, and S. Misra, "Assessment of the suitability of fog computing in the context of Internet of Things," *IEEE Trans. Cloud Comput.*, vol. 6, no. 1, pp. 46–59, Mar. 2018.

[16] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1171–1181, Dec. 2016.

[17] S. Vakilian and A. Fanian, "Enhancing users' quality of experienced with minimum energy consumption by fog nodes cooperation in Internet of Things," in *Proc. 28th Iranian Conf. Electr. Eng. (ICEE)*, Aug. 2020, pp. 1–5.

[18] S. Vakilian, S. V. Moravvej, and A. Fanian, "Using the cuckoo algorithm to optimizing the response time and energy consumption cost of fog nodes by considering collaboration in the fog layer," in *Proc. 5th Int. Conf. Internet Things Appl. (IoT)*, May 2021, pp. 1–5.

[19] Y. Dong, S. Guo, J. Liu, and Y. Yang, "Energy-efficient fair cooperation fog computing in mobile edge networks for smart city," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7543–7554, Oct. 2019.

[20] P. Cai, F. Yang, J. Wang, X. Wu, Y. Yang, and X. Luo, "JOTE: Joint offloading of tasks and energy in fog-enabled IoT networks," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3067–3082, Apr. 2020.

[21] B. Kopras, F. Idzikowski, and P. Kryszkiewicz, "Power consumption and delay in wired parts of fog computing networks," in *Proc. IEEE Sustainability Through ICT Summit (StICT)*, Montreal, QC, Canada, Jun. 2019, pp. 1–8.

[22] Y. Wang, T. Zhao, L. Li, Z. Hou, and J. Gu, "Roofline model based performance-aware energy management for scientific computing," in *Proc. 9th Int. Symp. Parallel Architectures, Algorithms Program. (PAAP)*, Dec. 2018, pp. 74–80.

[23] N. M. Allayla and S. A. Dawwd, "Performance optimization on GPGPU & multicore CPU using roofline model," *Proc. IOP Conf., Mater. Sci. Eng.*, May 2021, vol. 1152, no. 1, Art. no. 012021.

[24] T. N. Gia, M. Jiang, A.-M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen, "Fog computing in healthcare Internet of Things: A case study on ECG feature extraction," in *Proc. IEEE Int. Conf. Comput. Inf. Technol.; Ubiquitous Comput. Commun.; Dependable, Autonomic Secure Comput.; Pervasive Intell. Comput.*, Oct. 2015, pp. 356–363.

[25] E. Strohmaier, J. Dongarra, H. Simon, and M. Martin. *Green500 list for November 2020*. Accessed: Dec. 2, 2021. [Online]. Available: https://www.top500.org/lists/green500/2020/11/

[26] R. Dolbeau, "Theoretical peak FLOPS per instruction set: A tutorial," *J. Supercomput.*, vol. 74, no. 3, pp. 1341–1377, Mar. 2018.

[27] S. Park et al., "Accurate modeling of the delay and energy overhead of dynamic voltage and frequency scaling in modern microprocessors," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 32, no. 5, pp. 695–708, May 2013.

[28] M. Olbrich, F. Nadolni, F. Idzikowski, and H. Woesner, "Measurements of path characteristics in PlanetLab," TU Berlin, Berlin, Germany, Tech. Rep. TKN-09-005, Jul. 2009.

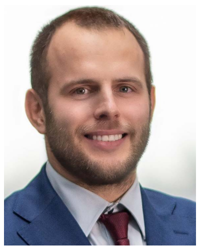
[29] L. Liberti, "Undecidability and hardness in mixed-integer nonlinear programming," *RAIRO-Oper. Res.*, vol. 53, no. 1, pp. 81–109, 2018.

[30] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[31] B. Bossy, P. Kryszkiewicz, and H. Bogucka, "Energy efficient wireless relay networks with computational awareness," *IEEE Trans. Commun.*, vol. 68, no. 2, pp. 825–840, Feb. 2020.
[32] B. Bossy, P. Kryszkiewicz, and H. Bogucka, "Energy efficient resource allocation in multiuser DF relay interference networks," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2018, pp. 1–6.
[33] T. Wang and L. Vandendorpe, "Successive convex approximation based methods for dynamic spectrum management," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2012, pp. 4061–4065.
[34] D. P. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1439–1451, Aug. 2006.
[35] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics Quart.*, vol. 2, nos. 1–2, pp. 83–97, 1955.
[36] H. Wong. (Oct. 2012). *A Comparison of Intel's 32 nm and 22 nm Core i5 CPUs: Power, Voltage, Temperature, and Frequency*. Accessed: Dec. 2, 2021. [Online]. Available: http://blog.stuffedcow.net/2012/10/intel32nm-22nm-core-i5-comparison/
[37] Intel. (Nov. 2012). *Intel Delivers New Architecture for Discovery With Intel Xeon Phi Coprocessor*. Accessed: Dec. 2, 2021. [Online]. Available: https://newsroom.intel.com/news-releases/intel-delivers-new-architecture-for-discovery-with-intel-xeon-phi-coprocessors/#gs.721jg4jg
[38] IEEE, *IEEE Standard for Information Technology–Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks–Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, Standard P802.11-REVmd/D2.0, 2020, p. 4379. [Online]. Available: https://ieeexplore.ieee.org/document/9363693
[39] O. Sharon and Y. Alpert, "The combination of QoS, aggregation and RTS/CTS in very high throughput IEEE 802.11ac networks," *Phys. Commun.*, vol. 15, pp. 25–45, Jun. 2015.
[40] S. Ivanov, D. Botvich, and S. Balasubramaniam, "On delay distribution in IEEE 802.11 wireless networks," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2011, pp. 254–256.
[41] G. Z. Khan, R. Gonzalez, E.-C. Park, and X.-W. Wu, "Analysis of very high throughput (VHT) at MAC and PHY layers under MIMO channel in IEEE 802.11ac WLAN," in *Proc. 19th Int. Conf. Adv. Commun. Technol. (ICACT)*, 2017, pp. 877–888.
[42] L. Uryvsky, A. Moshynska, and S. Osypchuk, "Applied research of modulation-coding schemes selection algorithms effectiveness in 802.11 equipment," in *Proc. 4th Int. Sci.-Practical Conf. Problems Infocommun. Sci. Technol. (PIC S&T)*, Oct. 2017, pp. 405–409.
[43] P. Bertoldi, "EU code of conduct on energy consumption of broadband equipment: Version 6," Joint Res. Centre, Ispra, Italy, Tech. Rep. JRC106039, 2017.
[44] W. Van Heddeghem, F. Idzikowski, W. Vereecken, D. Colle, M. Pickavet, and P. Demeester, "Power consumption modeling in optical multilayer networks," *Photonic Netw. Commun.*, vol. 24, no. 2, pp. 86–102, 2012.
[45] A. Zappone, E. Björnson, L. Sanguinetti, and E. Jorswieck, "Globally optimal energy-efficient power control and receiver design in wireless networks," *IEEE Trans. Signal Process.*, vol. 65, no. 11, pp. 2844–2859, Jun. 2017.

Bartosz Kopras received the B.Sc. and M.Sc. degrees in telecommunications from the Poznan University of Technology (PUT) in 2019 and 2020, respectively, where he is currently pursuing the Ph.D. degree. His research interests include green communications, fog computing, and number theory. He was awarded the Scholarship of the Minister of Science and Higher Education in 2019.

Bartosz Bossy received the M.Sc. and Ph.D. degrees (Hons.) in telecommunications from the Poznan University of Technology (PUT) in 2015 and 2022, respectively. He is currently an Assistant Professor with the PUT. He has been involved in a number of national and international projects. His research interests include green communications, energy-efficient resource allocation, and fog networks and optimization.

Filip Idzikowski (Member, IEEE) received the M.S. degree in telecommunication engineering from the Poznan University of Technology (PUT), Poland, and Dublin City University, Ireland, and the Ph.D. degree from the Technical University of Berlin, Germany. He spent half a year at the University of Rome Sapienza, Italy. He is currently an Assistant Professor with the PUT. His research interests include the power consumption of multi-layer core networks, traffic modeling, protection, and routing. He regularly serves as a member for various committees on several conferences, including IEEE ICC and IEEE GLOBECOM. His Ph.D. thesis was awarded the German KuVS Prize.



Paweł Kryszkiewicz (Senior Member, IEEE) received the Ph.D. degrees in telecommunications from the Poznan University of Technology (PUT), Poland, in 2015 and 2022, respectively. He is currently an Assistant Professor with the Institute of Radiocommunications, PUT. He is a Fulbright Alumnus of the Worcester Polytechnic Institute, Worcester, MA, USA. He has been involved in a number of national and international (mostly EU-funded) research projects. He is the author of over 100 research articles, two book chapters, and one book. His research interests include multicarrier system design, green communications, fog computing, dynamic spectrum access, and massive MIMO systems.

Hanna Bogucka (Senior Member, IEEE) is currently a Full Professor and the Director of the Institute of Radiocommunications, Poznan University of Technology. She is the Co-Founder of RIMEDO Laboratories, a spin-off from the PUT. She is the author of 200 articles, handbooks, and scientific monographs on wireless communication and cognitive radio. Her research interests include wireless communications, radio resource management, and cognitive and green communication. She is a member of the Polish Academy of Sciences. She serves the IEEE ComSoc Fog/Edge Industry Community as an European Chair and as a member of the IEEE ComSoc Board of Governors representing EMEA Region.